02/16/05

Attorney's Docket No.: 07575-034001 / P01-1916.01

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant	: Bhanjois, et al.	Art Unit	: 2127
Serial No.	: 09/408,149	Examiner	: Ali, Syed J.
Filed	: September 29, 1999		
Title	: MICROKERNEL FOR REAL TIME APPLICATIONS		

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

BRIEF ON APPEAL

(1) Real Party in Interest

The real party in interest is Network Appliance, Incorporated. An assignment from the inventors to Auspex Systems, Incorporated was recorded in the U.S. Patent Office on September 29, 1999, Reel 010282, Frame 0482. A subsequent assignment from Auspex Systems, Incorporated to Network Appliance, Incorporated was recorded in the U.S. Patent Office on August 29, 2003, Reel 013933, Frame 0292.

(2) Related Appeals and Interferences

There are no related appeals or interferences known to the appellant.

(3) Status of Claims

Claims 1, 4-11, 14-21, 24-31, and 33 are pending.

Claims 1, 6-11, 16-21, and 26-30 stand rejected under 35 U.S.C. § 102(b) as being anticipated by U.S. Patent No. 5,742,825 to Mathur et al. ("Mathur").

Claims 4, 14, and 24 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Mathur.

Claims 5, 15, 25, 31 and 33 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Mathur in view of U.S. Patent No. 5,485,579 to Hitz et al. ("Hitz").

CERTIFICATE OF MAILING BY EXPRESS MAIL

Express Mail Label No. EV 584 752 195 US

February 14, 2005
Date of Deposit

(4) Status of Amendments

There are no unentered amendments.

(5) Summary of Claimed Subject Matter

A conventional operating system, e.g., a UNIX operating system, is logically layered and divided into two main portions: a kernel and user programs. Specification, page 2, line 8-9. The kernel interfaces with and controls the hardware, and also provides the user programs with a set of abstract system services called system calls. Specification, page 2, line 8-10. The kernel runs at a kernel level, where the kernel can execute privileged operations that allow the kernel to have full control over the hardware as well as user programs running at a user level. Specification, page 2, line 10-12. Such centralization provides an environment in which all user programs share the underlying hardware in a coordinated fashion. Specification, page 2, line 12-13.

The kernel typically schedules only processes for execution since all system activities, whether user or kernel level, occur within the context of some process. Specification, page 3, line 1-3. When using conventional time-sharing scheduling policies, processes executing at the user level may be time sliced at any time so that processing resources may be shared fairly among all processes. Specification, page 3, line 3-5. Processes operating at the kernel level, however, are exempt from time slicing. A switch to a different process while executing at the kernel level is typically performed only when the current kernel process explicitly allows the switch to occur. Specification, page 3, line 5-7.

In a conventional microkernel, certain services are implemented to run outside of the kernel at the user level in special server processes. Typically, the microkernel performs only inter-process communication (IPC) and process scheduling. External processes then use these core services to implement the remainder of the operating system functionally. The removal of complexity from the kernel allows a more efficient IPC implementation, that reduces the performance penalty incurred (from communicating with external service-providing processes) such that the microkernel can be comparable in performance to a monolithic kernel. Specification, page 2, line 17-23.

Claim 1 is directed to an operating system. The operating system includes a non-preemptive microkernel that executes two or more processes in accordance with a non-preemptive scheduling scheme, where each process executed by the non-preemptive microkernel is only interrupted for a higher priority process to execute when the process blocks or explicitly requests to be preempted. Specification, page 8, lines 26-27; Figure 1 and associated text. The operating system also includes one or more kernels that are each executed as a process by the non-preemptive microkernel. Specification, page 5 line 29 – page 6, line 1. At least one of these kernels executes an operating system as a dependent process, where the operating system is a time-sliced operating system or a time-sliced microkernel. Specification, page 6, line 27 – page 7, line 1; page 3, lines 19-21; Figure 3 and associated text.

Independent claim 11 recites a method for operating a computer system. The method includes managing two or more processes with a non-preemptive microkernel. The microkernel executes the two or more processes in accordance with a non-preemptive scheduling scheme, where each process executed by the non-preemptive microkernel is only interrupted for a higher priority process to execute when the process blocks or explicitly requests to be preempted. Specification, page 8, lines 26-27; Figure 1 and associated text. The method further includes executing one or more kernels as one or more processes managed by the non-preemptive microkernel, and executing an operating system in one of the one or more kernels as a dependent process. Specification, page 6, line 27 – page 7, line 1; Figure 3 and associated text. The operating system being executed is a time-sliced operating system or a time-sliced microkernel. Specification, page 3, lines 19-21.

Independent claim 21 is directed to a computer system. The computer system includes means for managing two or more processes with a non-preemptive microkernel. Figures 10-12 and associated text. The microkernel executes the two or more processes in accordance with a non-preemptive scheduling scheme, where each process executed by the non-preemptive microkernel is only interrupted for a higher priority process to execute when the process blocks or explicitly requests to be preempted. Specification, page 8, lines 26-27; Figure 1 and associated text. The computer system further includes means for executing one or more kernels as one or more processes managed by the non-preemptive microkernel, and means for executing an operating system in one of the one or more kernels as a dependent process. Figures 10-12 and

associated text. The operating system being executed is a time-sliced operating system or a time-sliced microkernel. Specification, page 6, line 27 – page 7, line 1; page 3, lines 19-21.

Independent claim 31 is directed to a computer. The computer includes an interconnect bus and one or more processors coupled to the interconnect bus. Figures 10-11 and associated text. The one or more processors are adapted to be configured for server specific functionalities including network processing, file processing, storage processing and application processing. Specification, page 14, lines 2-12. The computer further includes a configuration processor coupled to the interconnect bus and to the processors. The configuration processor dynamically assigns processor functionalities upon request. The computer also has one or more data storage devices coupled to the processors and managed by a file system. Figures 10-11 and associated text. The computer further includes a non-preemptive microkernel that executes two or more processes in accordance with a non-preemptive scheduling scheme, where each process executed by the non-preemptive microkernel is only interrupted for a higher priority process to execute when the process blocks or explicitly requests to be preempted. Specification, page 8, lines 26-27. The computer further includes one or more kernels each being executed as a process by the non-preemptive microkernel, where at least one of the one or more kernels executes an operating system as a dependent process. Specification, page 6, line 27 – page 7, line 1; Figure 3 and associated text. The operating system being executed is a time-sliced operating system or a time-sliced microkernel. Specification, page 6, line 27 – page 7, line 1; page 3, lines 19-21.

An advantage of the methods and apparatus of the claimed invention is that real-time applications such as multimedia streaming can be supported without allowing other operations to disrupt the capture, delivery or playback of data. No modification to an operating system's scheduling algorithm is needed. Moreover, the operating system applications that are running as processes are protected without degrading a real-time response capability of the operating system.

The non-preemptive microkernel protects the nature of the other microkernels or operating systems that the non-preemptive microkernel runs as processes. For example, a user can run a UNIX operating system as a process and schedule UNIX to run to protect the nature of all applications that are running on UNIX. When the UNIX operating system gets control of the

computer, the operating system can run applications to generate network or file system or storage tasks.

The non-preemptive microkernel offers scalability: simply by including or excluding additional microkernel processes, the functionality (and resource requirements) of the operating system can be scaled to address different application needs requiring different operating systems using the same microkernel. The non-preemptive microkernel also offers extensibility achieved by adding specific operating system microkernels running as processes. Moreover, these functionality enhancements can be readily accomplished by users, rather than requiring (or waiting for) a hardware vendor to implement them.

(6) Issues

1. Are claims 1, 6-11, 16-21, and 26-30 properly rejected under 35 U.S.C. § 102(b) as being anticipated by Mathur.

2. Are claims 4, 14, and 24 properly rejected under 35 U.S.C. § 103(a) as being unpatentable over Mathur.

3. Are claims 5, 15, 25, 31 and 33 properly rejected under 35 U.S.C. § 103(a) as being unpatentable over Mathur in view of Hitz.

(7) Argument

1. Claims 1, 6-11, 16-21, and 26-30 are not properly rejected under 35 U.S.C. § 102(b) as being anticipated by Mathur.

(A) Claims 1, 6-11, 16-21, and 26-30

Claim 1 recites a non-preemptive microkernel executing two or more processes in accordance with a non-preemptive scheduling scheme. A time-sliced operating system or a time-sliced microkernel is executed as a process by the non-preemptive microkernel.

Mathur does not disclose the claimed subject matter.

Mathur discloses a graphical windowing operating system for an office machine that can run both windowing applications and real-time applications. The graphical windowing operating system includes a non-preemptive scheduler and a preemptive scheduler. Col. 3, lines 32-35. Windowing applications are grouped together as one schedulable process – called a "system"

process – and are scheduled non-preemptively with the non-preemptive scheduler. Col. 3, lines 36-40. The "system" process and real-time applications are, in turn, scheduled preemptively with the preemptive scheduler. Col. 3, lines 40-42. Such a grouping permits windowing applications to be scheduled non-preemptively for a portion of available processor time, and then be preempted by real time applications for another portion of processor time. Col. 3, lines 54-57.

(A)(i) Mathur does not disclose a non-preemptive microkernel that executes a time-sliced operating system or a time-sliced microkernel as a process

Mathur discloses an office machine operating system (OS) kernel 40 that executes processes preemptively. Col. 15, lines 65-67, col. 16, lines 7-9. Processes executed by the office machine OS kernel 40 include a "system" process and one or more background processes. Col. 16, lines 7-9. Processes within the "system" process are executed non-preemptively. Col. 16, lines 5-7. Background processes are executed preemptively. Col. 12, lines 51-53. While Mathur may disclose non-preemptive kernel ("system process") executed as a process by a preemptive kernel (office machine OS kernel 40), and a preemptive kernel (background processes) executed as a process by a preemptive kernel (office machine OS kernel 40), Mathur does not disclose a time-sliced operating system or a time-sliced microkernel being executed as a process of a non-preemptive microkernel. Thus, Mathur does not disclose or teach a non-preemptive microkernel that executes a time-sliced operating system or a time-sliced microkernel as a process.

(A)(ii) The Examiner has not met the basic criteria to establish anticipation

To anticipate a claim, the reference must teach every element of the claim. "A claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference." *Verdegaal Bros. v. Union Oil Co. of California*, 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987). "The identical invention must be shown in as complete detail as is contained in the ... claim." *Richardson v. Suzuki Motor Co.*, 868 F.2d 1226, 1236, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989).

As discussed above, Mathur fails to disclose a non-preemptive microkernel that executes a time-sliced operating system or a time-sliced microkernel as a process and, therefore, Mathur

does not anticipate claim 1. Thus, claim 1 is improperly rejected under § 102(b). Claims 4-10 depend directly or indirectly from claim 1 and, therefore, are improperly rejected for at least the same reasons. Claims 11 and 21 recite features corresponding to those of claim 1 and, therefore, are also improperly rejected for at least the same reasons. Claims 16-20 and 26-30 respectively depend directly or indirectly from claims 11 and 21 and, therefore, are improperly rejected for at least the same reasons.

2. Claims 4, 14, and 24 are not properly rejected under 35 U.S.C. § 103(a) as being unpatentable over Mathur.

(A) Claims 4, 14, and 24

Claim 4 depends directly from claim 1, and recites that the operating system being executed by the one or more kernels is UNIX.

(A)(i) Mathur does not disclose a non-preemptive microkernel that executes a time-sliced operating system or a time-sliced microkernel as a process

Putting aside the issue of whether Mathur discloses a UNIX operating system, as discussed in Sections (1)(A)(i)-(ii), Mathur fails to disclose a non-preemptive microkernel that executes a time sliced operating system or a time sliced microkernel as a process. The appellant respectfully submits that claim 4 is improperly rejected for at least reasons similar to those discussed in Sections (1)(A)(i)-(ii) above.

(A)(ii) The Examiner has not met the basic criteria required to establish a *prima facie* case of obviousness

To establish a *prima facie* case of obviousness, the Examiner must make three basic showings. First, there must be some suggestion or motivation, either in the references or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable

expectation of success must both be found in the prior art, and not based on applicant's disclosure. *In re Vaeck*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991).

The Examiner has failed to make the three basic showings and consequently, no *prima facie* case of obviousness has been established. That is, Mathur fails to disclose a non-preemptive microkernel that executes a time sliced operating system or a time sliced microkernel as a process, as required by claim 4. Thus, claim 4 is improperly rejected under § 103(a). Claims 14 and 24 incorporate the features of claim 4 and are, therefore, improperly rejected for at least the same reasons.

3. Claims 5, 15, 25, 31, and 33 are not properly rejected under 35 U.S.C. § 103(a) as being unpatentable over Mathur in view of Hitz.

(A) Claims 5, 15, and 25

Claim 5 depends directly from claim 1, and recites that each of the two or more processes executed by the non-preemptive microkernel has its own stack.

(A)(i) Neither Mathur nor Hitz discloses a non-preemptive microkernel that executes a time-sliced operating system or a time-sliced microkernel as a process

The Examiner acknowledges that Mathur does not disclose processes that have their own stack. However, the Examiner asserts that Hitz "provides a framework for each process to have its own execution stack".

Even assuming that Hitz does provides a framework for each process to have its own execution stack, the Examiner has not shown that Hitz discloses a non-preemptive microkernel that executes a time-sliced operating system or a time-sliced microkernel as a process.

Neither Mathur nor Hitz discloses a non-preemptive microkernel that executes a time-sliced operating system or a time-sliced microkernel as a process as required by claim 5. Thus, Mathur and Hitz, either alone or in combination, do not disclose or teach a non-preemptive microkernel that executes a time-sliced operating system or a time-sliced microkernel as a process.

(A)(ii) The Examiner has not met the basic criteria required to establish a *prima facie* case of obviousness

To establish a *prima facie* case of obviousness, the Examiner must make the three basic showings described in Section (2)(A)(ii). Because neither Mathur nor Hitz teaches or suggests all the limitations of claim 5, the Examiner has failed to make the three basic showings and consequently, no *prima facie* case of obviousness has been established. Thus claim 5 is improperly rejected under § 103(a).

Claims 15 and 25 incorporate the features of claim 5 and, therefore, are also improperly rejected for at least the same reasons.

(B) Claim 31

Claim 31 recites a computer system that includes a non-preemptive microkernel that executes a time-sliced operating system or a time-sliced microkernel as a process.

(B)(i) Neither Mathur nor Hitz discloses a non-preemptive microkernel that executes a time-sliced operating system or a time-sliced microkernel as a process

The appellant respectfully submits that claim 31 is improperly rejected for at least reasons similar to those discussed in Sections (3)(A)(i)- (ii) above.

(C) Claim 33

Claim 33 incorporates the features of claim 31, and further recites that the non-preemptive microkernel executes a network switch operating system as a dependent process.

(C)(i) Neither Mathur nor Hitz discloses a non-preemptive microkernel that executes a time-sliced operating system or a time-sliced microkernel as a process

The appellant respectfully submits that claim 31 is improperly rejected for at least reasons similar to those discussed in Sections (3)(A)(i)-(ii) above.

Conclusion

Neither Mathur nor Hitz discloses a non-preemptive microkernel that executes a time-sliced operating system or a time-sliced microkernel as a process, as required by the claims. The appellant, therefore, respectfully submits that the pending claims 1, 4-11, 14-21, 24-31, and 33 are not properly rejected under § 102 or § 103.

The brief fee of $250 is enclosed. Please apply any other charges or credits to Deposit Account No. 06-1050.

Respectfully submitted,

Date: 02-14-05

Kelvin M. Vivian
Reg. No. 53,727

Fish & Richardson P.C.
500 Arguello Street, Suite 500
Redwood City, California 94063
Telephone: (650) 839-5070
Facsimile: (650) 839-5071

50261206.doc

Appendix of Claims

1. (Previously Presented) An operating system, comprising:

a non-preemptive microkernel executing two or more processes in accordance with a non-preemptive scheduling scheme, wherein each process executed by the non-preemptive microkernel is only interrupted for a higher priority process to execute when the process blocks or explicitly requests to be preempted; and

one or more kernels each being executed as a process by the non-preemptive microkernel,

wherein at least one of the one or more kernels executes an operating system as a dependent process, the operating system being a time-sliced operating system or a time-sliced microkernel.

2 – 3. (Cancelled)

4. (Previously Presented) The operating system of claim 1, wherein the operating system is Unix.

5. (Previously Presented) The operating system of claim 1, wherein each of the two or more processes executed by the non-preemptive microkernel has its own stack.

6. (Previously Presented) The operating system of claim 1, wherein each of the two or more processes executed by the non-preemptive microkernel communicate using one or more messages.

7. (Previously Presented) The operating system of claim 1, wherein each of the two or more processes executed by the non-preemptive microkernel has a unique process identifier (PID).

8. (Original) The operating system of claim 7, further comprising a mailbox coupled to a plurality of processes to service messages sent to a single PID.

9. (Previously Presented) The operating system of claim 1, wherein each of the two ore more processes executed by the non-preemptive microkernel never terminates.

10. (Previously Presented) The operating system of claim 1, wherein one of the one or more kernels is a microkernel.

11. (Previously Presented) A method for operating a computer system including a CPU, comprising:

managing two or more processes with a non-preemptive microkernel, the microkernel executing the two or more processes in accordance with a non-preemptive scheduling scheme, wherein each process executed by the non-preemptive microkernel is only interrupted for a higher priority process to execute when the process blocks or explicitly requests to be preempted;

executing one or more kernels as one or more processes managed by the non-preemptive microkernel; and

executing an operating system in one of the one or more kernels as a dependent process, the operating system being a time-sliced operating system or a time-sliced microkernel.

12 – 13. (Cancelled)

14. (Previously Presented) The method of claim 11, wherein the operating system is Unix.

15. (Previously Presented) The method of claim 11, wherein each of the two or more processes executed by the non-preemptive microkernel has its own stack.

16. (Original) The method of claim 11, further comprising performing inter-process communication using one or more messages.

17. (Previously Presented) The method of claim 11, wherein each of the two or more processes executed by the non-preemptive microkernel has a unique process identifier (PID).

18. (Previously Presented) The method of claim 17, further comprising servicing messages sent to a single PID by a plurality of processes using a mailbox.

19. (Previously Presented) The method of claim 11, further comprising executing the two or more processes without termination.

20. (Previously Presented) The method of claim 11, further comprising executing a microkernel in one of the one or more kernels.

21. (Previously Presented) A computer system, comprising:

means for managing two or more processes with a non-preemptive microkernel, the microkernel executing the two or more processes in accordance with a non-preemptive scheduling scheme, wherein each process executed by the non-preemptive microkernel is only interrupted for a higher priority process to execute when the process blocks or explicitly requests to be preempted;

means for executing one or more kernels as one or more processes managed by the non-preemptive microkernel; and

means for executing an operating system in one of the one or more kernels as a dependent process, the operating system being a time-sliced operating system or a time-sliced microkernel.

22 – 23. (Cancelled)

24. (Previously Presented) The system of claim 21, wherein the operating system is Unix.

25. (Previously Presented) The system of claim 21, wherein each of the two or more processes executed by the non-preemptive microkernel has its own stack.

26. (Original) The system of claim 21, further comprising means for performing inter-process communication using one or more messages.

27. (Previously Presented) The system of claim 21, wherein each of the two or more processes executed by the non-preemptive microkernel has a unique process identifier (PID).

28. (Previously Presented) The system of claim 21, further comprising means for servicing messages sent to a single PID by a plurality of processes using a mailbox.

29. (Previously Presented) The system of claim 21, further comprising means for executing each of the two or more processes executed by the non-preemptive microkernel without termination.

30. (Previously Presented) The system of claim 21, further comprising means for executing a microkernel in one of the one or more kernels.

31. (Previously Presented) A computer, comprising:

an interconnect bus;

one or more processors coupled to the interconnect bus and adapted to be configured for server-specific functionalities including network processing, file processing, storage processing and application processing;

a configuration processor coupled to the interconnect bus and to the processors, the configuration processor dynamically assigning processor functionalities upon request;

one or more data storage devices coupled to the processors and managed by a file system;

a non-preemptive microkernel executing two or more processes in accordance with a non-preemptive scheduling scheme, wherein each process executed by the non-preemptive microkernel is only interrupted for a higher priority process to execute when the process blocks or explicitly requests to be preempted; and

one or more kernels each being executed as a process by the non-preemptive microkernel,

wherein at least one of the one or more kernels executes an operating system as a dependent process, the operating system being a time-sliced operating system or a time-sliced microkernel.

32. (Cancelled)

33. (Previously Presented) The computer of claim 31, wherein the non-preemptive microkernel executes a network switch operating system as a dependent process.